

APPENDIX A

```
<!--

/*****
 *
 * U.S. Patent Pending. Copyright 1999, 2000 Yahoo! Inc.,
 * 3420 Central Expressway, Santa Clara, California U.S.A.
 *
 * ALL RIGHTS RESERVED.
 *
 * This computer program is protected by copyright law and
 * international treaties. Unauthorized reproduction or
 * distribution of this program, or any portion of it, may
 * result in severe civil and criminal penalties, and will
 * be prosecuted to the maximum extent possible under the law.
 *
 *****/

/*****
 * Exception handling
 *****/
function rmi_handleError (err, url, line)
{
    // alert('BAD: \n' + err + '\n' + url + '\nline no: ' + line);
    // window.status = 'BAD: \n' + err + '\n' + url + '\nline no: ' + line;
    window.status = "Javascript: Done (" + line + ")";

    return true;           // error is handled
}
window.onerror = rmi_handleError;

/*****
 * Globals
 *****/
var rmi_Vars = "/rmivars%3ftarget=_top";
var rmi_FramesetTagCounter = 0;
var rmi_CookieDomain = ".yahoo.com";

// delete the 1st character, <.
rmi_FrameWrapper = rmi_FrameWrapper.substring(1, rmi_FrameWrapper.length);

/*****
 * Translate a string, then write to the browser.
 *****/

function rmi_writeln(obj, str)
{
    var newStr;

    if (arguments.length == 2) {
        newStr = rmi_xlate(str);

        if (obj == document && (typeof document.layers != "undefined")
            && (typeof document.layers['rmilayer'] != "undefined")) )
```

```

        document.layers['rmilayer'].document.writeln(newStr);
    else
        obj.writeln(newStr);
} else {
    newStr = rmi_xlate(obj); // for backward compatibility with hseds
    document.writeln(newStr);
}
}

```

```

function rmi_write(obj, str)
{
    var newStr;

    if (arguments.length == 2) {
        newStr = rmi_xlate(str);

        if (obj == document && (typeof document.layers != "undefined")
            && (typeof document.layers['rmilayer'] != "undefined") )
            document.layers['rmilayer'].document.write(newStr);
        else
            obj.write(newStr);
    } else {
        newStr = rmi_xlate(obj); // for backward compatibility with hseds
        document.write(newStr);
    }
}

```

```

/*****
 * String utilities
 *****/

```

```

function rmi_startsWith(full, sub)
{
    var fullLower = full.toLowerCase();
    var subLower = sub.toLowerCase();
    var index = fullLower.indexOf(subLower);
    return index ? false : true;
}

```

```

function rmi_endsWith(full, sub)
{
    var fullLower = full.toLowerCase();
    var subLower = sub.toLowerCase();
    var offset = fullLower.length - subLower.length;
    if (offset < 0) return false;

    var index = fullLower.indexOf(subLower, offset);
    return (index==offset) ? true : false;
}

```

```

function rmi_endsExactlyWith(full, sub)
{
    var offset = full.length - sub.length;
    if (offset < 0) return false;

    var index = full.indexOf(sub, offset);
    return (index==offset) ? true : false;
}

```

```

}

/* gets the port of an URL */
function rmi_getPort(url)
{
    var host = rmi_getHost(url);          // get "host:port"
    var begin = host.indexOf(":");

    if (begin == -1 || (host.length - begin) < 2)    // e.g. length of ':80'
    {
        return (rmi_getProtocol(url) == "https") ? "443" : "80" ;
    }
    else
    {
        return host.substring(begin+1, host.length);    // +1 for ':'
    }
}

/* gets the protocol of an URL */
function rmi_getProtocol(url)
{
    var index = url.indexOf("://");
    return url.substring(0, index);
}

/* http://HOST/whatever
 * return HOST
 */
function rmi_getHost(url)
{
    var end = url.indexOf("://");
    var next = end + 3;

    end = url.indexOf("/", next);
    if (end == -1) end = url.length;
    return url.substring(next, end);
}

/* http://HOSTNAME:port/whatever
 * return HOSTNAME
 */
function rmi_getHostname(url)
{
    var host = rmi_getHost(url);          // get "host:port"
    var index = host.indexOf(":");

    if (index == -1)
        return host;
    else
        return host.substring(0, index);
}

/* http://host/FILE
 * return FILE
 */
function rmi_getFile(url)
{

```

```

var end = url.indexOf("://");
var next = end + 3;

end = url.indexOf("/", next);

if (end == -1)
    return "/";
else
    return url.substring(end, url.length);
}

/* PROTOCOL://HOST/FILE
 * return URLRoot == PROTOCOL://HOST
 */
function rmi_getURLRoot(url)
{
    var protocol = rmi_getProtocol(url);
    var host = rmi_getHost(url);
    return protocol + "://" + host
}

function rmi_dirname(full)
{
    var dir = full;

    // Remove cgi parameters (e.g. "?k1=v1&k2=v2...")
    // Because the parameters might have '/'

    var ind = dir.indexOf('?');
    if ( ind >= 0 ) dir = dir.substring(0, ind);

    ind = dir.lastIndexOf('/');    // search from right
    if (ind == -1) return "";      // no slash
    if (ind == 0) return "/";     // root

    if ( rmi_endsExactlyWith(dir.substring(0, ind+1), "://") )
        ind = dir.length;

    return dir.substring(0, ind);
}

/* Trim leading & ending quotes
 */
function rmi_trimQuotes(str)
{
    var first = str.charAt(0);
    if (first == '"') return (str.substring(1, str.length-1));
    if (first == '\') return (str.substring(1, str.length-1));
    return str;
}

/* Trim leading & ending spaces
 */
function rmi_trim(str)
{
    if (typeof str == "undefined") return "";

```

```

var start = 0;
var end = str.length;

for (var i=0; i < str.length; ++i)
{
    if (str.charAt(i) == ' ') continue;

    start = i;
    break;
}

for (var i=str.length-1; i >= 0 ; --i)
{
    if (str.charAt(i) == ' ') continue;

    end = i + 1;
    break;
}

return str.substring(start, end);
}

/*****
 * Normalize URL
 *****/
function rmi_normalizeURL(in_url)
{
    var url = in_url.toString();
    var first = url.charAt(0);
    if (first == '"') url = url.substring(1, url.length-1);

    var ret = url;

    if ( rmi_startsWith(url, "http://") )
    {
        ret = url;
    }
    else if ( rmi_startsWith(url, "https://") )
    {
        ret = url;
    }
    else if ( rmi_startsWith(url, "/" ) )
    {
        ret = rmi_getURLRoot(rmi_BaseURL) + url;
    }
    else if ( rmi_startsWith(url, "#" ) )
    {
        ret = document.location + url;
    }
    else // relative
    {
        var dir = rmi_dirname(rmi_BaseURL);
        ret = dir + "/" + url;
    }

    return ret;
}

```

```

/*****
 * Translate a URL (in the case of form action)
 * If the incoming code is the form of location=url
 * then we return location=rmi_xlateURL(url)
 *****/
function rmi_xlateAction(action_url)
{
    var ret = "";
    var url = action_url.toString();

    if ( rmi_startsWith(url, "location=") ) {
        var new_loc = url.substring(url.indexOf("=")+1, url.length);
        ret = "location=" + rmi_xlateURL(new_loc) + "'";
    } else {
        ret = rmi_xlateURL(url);
    }

    if (rmi_JsDebug.indexOf(",rmi_xlateAction,") != -1)
        alert("rmi_xlateAction: old: " + action_url + "\n" + "new: " + ret);

    return ( ret );
}

/*****
 * Translate a URL
 * Note: if url is already starts with rmi proxy url it will
 * not be translated again.
 *****/
function rmi_xlateURL(in_url)
{
    var ret = "";
    var url = in_url.toString();
    var first = url.charAt(0);
    if (first == '"') url = url.substring(1, url.length-1);
    if (first == '\') url = url.substring(1, url.length-1);

    // Ignore javascript:
    if ( rmi_startsWith(url, "javascript:") )
        return url;

    url = rmi_normalizeURL(url);

    if ( rmi_startsWith(url, rmi_ProxyURL) ||
        rmi_startsWith(url, rmi_SecureProxyURL) ||
        rmi_endsWith(url, ".jpg") ||
        rmi_endsWith(url, ".jpeg") ||
        rmi_endsWith(url, ".gif")
    )
    {
        return url;
    }

    /*
     * Collapse the file part of an URL

```

```

    */
    var urlroot = rmi_getURLRoot(url);
    var file = pathCollapse(rmi_getFile(url));

    ret = urlroot + file;

    if ( rmi_startsWith(url, "https://") )
        ret = rmi_SecureProxyURL + ret;
    else
        ret = rmi_ProxyURL + ret;

    if (rmi_FrameWrapperMode && rmi_UrlTarget == "_top")    // onTop & wrapper
mode
        ret = rmi_appendToUrl(ret, rmi_Vars);

    if (rmi_JsDebug.indexOf(",rmi_xlateURL,") != -1)
        alert("rmi_xlateURL:\n" + "in_url: " + in_url + "\n" + "ret: " + ret +
"\n");

    return ret;
}

/*****
 * Get original (before RMI) location property (href, host, etc)
 *****/
function rmi_getOriginal(loc, prop)
{
    var origUrl = "" + loc;
    var url = "" + loc;
    var index = url.indexOf("/rmi/");
    var ret = "";

    if (index != -1)
        origUrl = url.substring(index+5, url.length);    // Get string after
"/rmi/"

    if (prop == "host")
        ret = rmi_getHostname(origUrl) + ":" + rmi_getPort(origUrl);
    else if (prop == "hostname")
        ret = rmi_getHostname(origUrl);
    else if (prop == "port")
        ret = rmi_getPort(origUrl);
    else if (prop == "pathname")
    {
        var path = rmi_getFile(origUrl);
        ret = (path.indexOf("?") == -1 ) ? path : path.substring(0,
path.indexOf("?"));
    }
    else if (prop == "search")
    {
        var path = rmi_getFile(origUrl);
        ret = (path.indexOf("?") == -1) ? "" : path.substring(path.indexOf("?"),
path.length);
    }
    else
        ret = origUrl;    // location, location.href, & all others

```

```

// Remove RMI var string (e.g. /rmivars%3f...).
// KEEP text before AND after RMI var strings

var rmiVarStr = "/rmivars";
var rmiVarStrLen = 9;          // length of "/rmivars?" for IE

var i_rmiVarStr = ret.indexOf(rmiVarStr);
var head = (i_rmiVarStr == -1) ? ret : ret.substring(0, i_rmiVarStr);

var tail = "";
if (i_rmiVarStr != -1)          // RMI var string exists
{
    var i1 = ret.indexOf("?", i_rmiVarStr + rmiVarStrLen);
    var i2 = ret.indexOf("#", i_rmiVarStr + rmiVarStrLen);

    if (i1 != -1) tail = ret.substring(i1, ret.length);
    else if (i2 != -1) tail = ret.substring(i2, ret.length);
}

return head + tail;
}

/*****
 * Get original (before RMI) document.domain property
 *****/
function rmi_getOriginalDomain()
{
    var origUrl = "" + window.location;
    var url = origUrl;
    var index = url.indexOf("/rmi/");
    var ret = "";

    if (index != -1)
        origUrl = url.substring(index+5, url.length);    // Get string after
"/rmi/"

    ret = rmi_getHostname(origUrl);

    // Remove RMI tail string (e.g. /rmivars%3f...)

    var rmiTail = "/rmivars%3f";
    ret = (ret.indexOf(rmiTail) == -1) ? ret : ret.substring(0,
ret.indexOf(rmiTail));

    return ret;
}

/*****
 * Return a frame object
 *****/
function rmi_getFrame(win, index)
{
    if (!rmi_FrameWrapperMode) return (win.frames[index]);

    // FrameWrapperMode from here on...

```

```

if ((typeof index) == "number")
{
    if (win == top)
        return(win.frames[index+1]);    // +1 due to Yahoo's extra frame
    else
        return(win.frames[index]);
}
else
    return(win.frames[index]);    // string & other types
}

/*****
* Get the current dimension of the RMI bar
*
* h = rmi_getBarDimensions("height")
*
* <FUTURE> w = rmi_getBarDimensions("width")
*
*****/
function rmi_getBarDimensions(dimType)
{
    var doc;

    var defaultRet = 50;

    if (top.frames.length == 0)    // non-frame site
        doc = document;
    else
    {
        // doc = top.frames[0].document;

        return defaultRet;
    }

    if ( window.navigator.appName.toLowerCase().indexOf("microsoft") != -1 )
    {
        // IE

        if (typeof doc.all.rmi_south_gif == "undefined")
            return defaultRet;
        else
        {
            if (dimType == "height")
                return doc.all.rmi_south_gif.offsetTop;
            else
                return defaultRet;
        }
    }
    else
    {
        // netscape

        if (typeof doc.rmi_south_gif == "undefined")
            return defaultRet;
        else
        {

```

```

        if (dimType == "height")
            return doc.rmi_south_gif.y;
        else
            return defaultRet;
    }
}

return defaultRet;          // no match (shouldn't be here)
}

/*****
 * Translate the options before opening a window (e.g. window.open)
 *****/
function rmi_xlateWinOpt(options)
{
    var tokens = options.split(",");

    var ret = "";
    for (var i=0; i<tokens.length; ++i)
    {
        var pair = tokens[i].split("=");
        var key = rmi_trim(pair[0]);
        var val = rmi_trim(pair[1]);

        if (key == "height")
        {
            var offset = rmi_getBarDimensions(key);

            if (val == "") val = "0";          // if no height value!
            val = "" + (parseInt(val) + offset);
        }

        if (i != 0) ret += ",";

        if (val == "")          // if no value
            ret += key;
        else
            ret += key + "=" + val;
    }

    return ret;
}

/*****
 * Open a window (using a window object from 1st argument)
 *****/
function rmi_winobj_open(winobj, url, target, options)
{
    //alert(url);
    //alert(target);
    //alert(options);

    var win;

    if (arguments.length == 2)
        win = winobj.open(rmi_xlateURL(url));
    else if (arguments.length == 3)

```

```

        win = winobj.open(rmi_xlateURL(url), target);
    else
        win = winobj.open(rmi_xlateURL(url), target, rmi_xlateWinOpt(options) );

    if (win != null) win.opener = self;

    return win;
}

/*****
 * Open a window (using a default window object)
 *****/
function rmi_window_open(url, target, options)
{
    //alert(url);
    //alert(target);
    //alert(options);

    var win;

    if (arguments.length == 1)
        win = window.open(rmi_xlateURL(url));
    else if (arguments.length == 2)
        win = window.open(rmi_xlateURL(url), target);
    else
        win = window.open(rmi_xlateURL(url), target, rmi_xlateWinOpt(options) );

    if (win != null) win.opener = self;

    return win;
}

function rmi_window_open_self(url)
{
    return window.open(rmi_xlateURL(url), "_self");
}

/*****
 * Get 'top' window for RMI
 *****/
function rmi_getTop(win)
{
    if (rmi_FrameWrapperMode) // frame wrapper
        return win;
    else if (top.frames.length > 1) // old frame
        return (win == top) ? top._rmi_bottom : win;
    else // non-frame
        return win;
}

/*****
 * Translate a target URL, then replace the document in
 * the target window
 *****/

```

```

function rmi_replace(win, url)
{
    if (win == "") win = self;

    if (rmi_FrameWrapperMode)
    {
        if (win == top)
            win.location.replace(rmi_xlateURL(url) + rmi_Vars);
        else
            win.location.replace(rmi_xlateURL(url));
    }
    else if (top.frames.length > 1)           // old frame mode
    {
        if (win == top)
            top._rmi_bottom.location.replace(rmi_xlateURL(url));
        else
            win.location.replace(rmi_xlateURL(url));
    }
    else                                       // non-frame mode
        win.location.replace(rmi_xlateURL(url));
}

/*****
 * Handle location setting for different modes after JS translation
 *
 * <Sample translation>
 * From: window.top.parent.location.href = url;
 * To: rmi_setLocation("window.top", ".parent.location.href",
rmi_xlateURL(url), window.top.parent);
 *****/
function rmi_setLocation(s1, s2, url, win)
{
    var frameName = "";
    var newUrl = url;

    if (rmi_FrameWrapperMode)
    {
        //@@ if (rmi_startsWith(s2, ". location"))
        if ( win == top )
        {
            frameName = "";
            newUrl = rmi_appendToUrl(url, rmi_Vars);
        }

        // Handle topmost frames

        var aWin = eval(s1);
        var array = s2.split(".");
        var head = rmi_trim(array[0]);

        if (aWin == top && rmi_startsWith(head, "frames" ))
        {
            var i0 = head.indexOf("[");
            var i1 = head.indexOf("]");

            var num = 0;
            num = head.substring(i0+1, i1);

```

```

        if (num >= 0)      // If a valid frame number, increment it
        {
            array[0] = "frames[" + (++num) + "]";
            s2 = array.join(".");
        }
    }
else                                // old mode
{
    frameName = "._rmi_bottom";
    newUrl = url
}

var code = s1 + frameName + "." + s2 + " = \"" + newUrl + "\";";
eval(code);

if (rmi_JsDebug.indexOf(",rmi_setLocation,") != -1)
    alert("rmi_setLocation:\n" + "url: " + url + "\n" + "code: " + code +
"\n");
}

/*****
 * Xlate a String
 *****/

/*
 * * If it returns a value different from str
 *   rmi_xlate will return new value.
 * else (i.e. rmi_xlate_merchant returns str)
 *   rmi_xlate will do regular processing
 */
function rmi_xlate_merchant(str)
{
    // alert("merchant dummy function");
    return str;
}

function rmi_xlate(pStr)
{
    var xlatedStr = "";
    var iSearch, iFrame, iImg, length, startLoc, endLoc;
    var offset1, offset2, head, src, tail;
    var str = "" + pStr;                // to string to be sure
    var lowercaseStr = str.toLowerCase();

    //
    // invoke merchant specific stuff
    //
    xlatedStr = rmi_xlate_merchant(str);
    if (xlatedStr != str) return xlatedStr;

    var parseStr = rmi_parseloop(str);
    if (parseStr != str) return parseStr;

```

```

// debugging block begin //////////////////////////////////////
//xlatedStr = rmi_xlate_src_href(str);
//if (parseStr != str) {
//  if (parseStr != xlatedStr) {
//    alert("parseStr " + parseStr + "\n xlatedStr " + xlatedStr);
//  }
//  return parseStr;
//}
// debugging block end //////////////////////////////////////

return(str);
}

function rmi_parseloop(str)
{
  var tagStr = str;
  var newStr = "";

  while (1) {
    var left, tag, right, nexttag;
    var l, r;

    l = tagStr.indexOf("<");

    // if there is no "<" sign, return tagStr
    if (l == -1) {
      newStr = newStr + tagStr;
      //alert("no < found in " + tagStr + "\n" + newStr);
      break;
    }
    left = tagStr.substring(0, l+1);

    r = tagStr.indexOf(">");
    // if there is no ">" sign, return tagStr
    if (r == -1) {
      newStr = newStr + tagStr;
      //alert("NO > found in " + tagStr + "\n" + newStr);
      break;
    }
    tag = tagStr.substring(l+1, r);

    nexttag = tagStr.indexOf("<", r);

    if (r < l) {
      // if " ... > .. <...>", then add upto < and
      // then loop back
      newStr = newStr + left;
      tagStr = tagStr.substring(l+1, tagStr.length);
    } else if (nexttag == -1) {
      right = tagStr.substring(r, tagStr.length);

      tag = rmi_xlate_src_href(tag);
      tag = rmi_xlate_form_action(tag);
      tag = rmi_xlate_frameset(tag); // do frameset last because
      extra tags are added

      if (rmi_FrameWrapperMode)

```

00650273 082900

```
        tag = rmi_doTargetInFrameWrapperMode(tag);
    else
        tag = rmi_xlate_target(tag);

    newStr = newStr + left + tag + right;
    break;
} else {
    right = tagStr.substring(r, nexttag);

    tag = rmi_xlate_src_href(tag);
    tag = rmi_xlate_form_action(tag);
    tag = rmi_xlate_frameset(tag);          // do frameset last because
extra tags are added

    if (rmi_FrameWrapperMode)
        tag = rmi_doTargetInFrameWrapperMode(tag);
    else
        tag = rmi_xlate_target(tag);

    newStr = newStr + left + tag + right;
    tagStr = tagStr.substring(nexttag, tagStr.length);

    // newStr = newStr + "_" + left + "#" + tag + "#" + right + "_";
    // loop back
}

}

if (str != "" && newStr == "") {
    newStr = str;
}

if (rmi_JsDebug.indexOf(",rmi_parseloop,") != -1)
    alert("parseloop:\n" + "old: " + str + "\n" + "new: " + newStr);

// debugging block begin //////////////////////////////////////
//var lowercaseStr = str.toLowerCase();
//if ((lowercaseStr.indexOf("src=") != -1 ||
//    lowercaseStr.indexOf("href=") != -1)
//    && lowercaseStr.indexOf("image ") == -1)
//    {
//        alert("orig " + str + "\npars " + newStr);
//    }
// debugging block end //////////////////////////////////////

return newStr;
}

function rmi_xlate_src_href(str)
{
    var newStr = "";
    var iSearch, iFrame, iImg, length, startLoc, endLoc;
    var offset1, offset2, head, src, tail;
    var lowercaseStr = str.toLowerCase();
```

```

iSearch = lowercaseStr.indexOf("src=");
if (iSearch != -1) {
    length = 4; // length of "src="

    // should not contain IMAGE tag
    iImg = lowercaseStr.indexOf("image ");
    if (iImg < iSearch && iImg > -1) return str;

    // should not contain IMG tag
    iImg = lowercaseStr.indexOf("img ");
    if (iImg < iSearch && iImg > -1) return str;

    iFrame = lowercaseStr.indexOf("frame");
    if (iFrame == -1) return str;
} else {
    iSearch = lowercaseStr.indexOf("href=");
    if (iSearch == -1) return str;

    // alert("found href in " + str);
    length = 5; // length of "href="
}

startLoc = iSearch + length;
head = str.substring(0, startLoc);

offset1 = lowercaseStr.indexOf(" ", startLoc);
if (offset1 == -1) {
    offset2 = lowercaseStr.indexOf(">", startLoc);
    if (offset2 == -1) {
        endLoc = str.length;
    } else {
        endLoc = offset2;
    }
} else {
    endLoc = offset1;
}

src = str.substring(startLoc, endLoc);
tail = str.substring(endLoc, str.length);

// Ignore 'javascript:*' & RETURN original string

if ( rmi_startsWith(src.toLowerCase(), "javascript:") ) return (str);
if ( rmi_startsWith(src.toLowerCase(), "javascript:") ) return (str);

var saved_urlTarget = rmi_UrlTarget; // saved to be restored
later
if (head.toLowerCase().indexOf("frame ") != -1)
    rmi_UrlTarget = ""; // <frame> will not be on
top

newStr = head + rmi_xlateURL(src) + tail;

rmi_UrlTarget = saved_urlTarget; // restore it

```

```

    if (rmi_JsDebug.indexOf(",rmi_xlate_src_href,") != -1)
        alert("rmi_xlate_src_href\n" + "old: " + str + "\n" + "new: " + newStr);

    return newStr;
}

function rmi_xlate_form_action(str)
{
    var lowercaseStr = str.toLowerCase();
    var iForm = lowercaseStr.indexOf("form");

    if (iForm == -1) return str;
    // alert (str);

    var iSearch = lowercaseStr.indexOf("action=");
    var length = 7; // length of "action="
    if (iSearch == -1) {
        iSearch = lowercaseStr.indexOf("action =");
        length = 9; // one more than length of string, to allow for extra space
    }
    if (iSearch == -1) return str;

    var startLoc, endLoc, offset1, offset2, head, src, tail;

    startLoc = iSearch + length;
    head = str.substring(0, startLoc);

    offset1 = lowercaseStr.indexOf(" ", startLoc);
    if (offset1 == -1) {
        offset2 = lowercaseStr.indexOf(">", startLoc);
        if (offset2 == -1) {
            endLoc = str.length;
        } else {
            endLoc = offset2;
        }
    } else {
        endLoc = offset1;
    }

    src = str.substring(startLoc, endLoc);
    tail = str.substring(endLoc, str.length);

    newStr = head + rmi_xlateURL(src) + tail;

    // alert(newStr);

    return newStr;
}

/*****
 * Write out the 'frame wrapper'
 *****/
function rmi_writeFrameWrapper()
{
    document.write(rmi_FrameWrapperText);
}

```

```

/*****
 * Handle a frameset tag (for top window in FrameWrapperMode ONLY)
 *****/
function rmi_xlate_frameset(tag)
{
    if (! rmi_FrameWrapperMode) return tag;
    if (self != top) return tag;

    var lowercaseStr = tag.toLowerCase();
    var iOpenTag = lowercaseStr.indexOf("frameset ");
    var iClosingTag = lowercaseStr.indexOf("/frameset");

    if (iOpenTag == -1 && iClosingTag == -1) return tag;    // not frameset tag

    if (iClosingTag >= 0)    // see </frameset>
    {
        -- rmi_FramesetTagCounter;

        // add Yahoo's </frameset> after the last frameset

        if (rmi_FramesetTagCounter == 0)
            ret = tag + ">/n</frameset";
        else
            ret = tag;
    }
    else    // see <frameset>
    {
        // add Yahoo's <frameset> before the 1st frameset

        if (rmi_FramesetTagCounter == 0)
            ret = rmi_FrameWrapper + "\n<" + tag;
        else
            ret = tag;

        ++ rmi_FramesetTagCounter;    // count <frameset> tag
    }

    if (rmi_JsDebug.indexOf(",rmi_xlate_frameset,") != -1)
        alert("rmi_xlate_frameset:\n" + "old: " + tag + "\n" + "new: " + ret);

    return (ret);
}

function rmi_xlate_target(str)
{
    var newStr = "";
    var iSearch, iFrame, iImg, length, startLoc, endLoc;
    var offset1, offset2, head, src, tail;
    var lowercaseStr = str.toLowerCase();
    var loc1, loc2, loc3;

    if (rmi_merchant_frames != "yes") {
        // alert("merchant frames not yes");
        return str;
    }
}

```

```

loc1 = lowercaseStr.indexOf("target=\"_top\"");
loc2 = lowercaseStr.indexOf("target=_top");
loc3 = lowercaseStr.indexOf("target='_top'");

if (loc1 != -1) {
    iSearch = loc1;
    length = 13;           // length of target="_top"
} else if (loc2 != -1) {
    iSearch = loc2;
    length = 11;           // length of target=_top
} else if (loc3 != -1) {
    iSearch = loc3;
    length = 13;           // length of target='_top'
} else {
    return str;
}

startLoc = iSearch;
endLoc = startLoc + length;

head = str.substring(0, startLoc);
src = "target=\"_rmi_bottom\"";
tail = str.substring(endLoc, str.length);

newStr = head + src + tail;

// alert("head " + head + "\nsrc= " + src + "\ntail " + tail);
// alert("str= " + str + "\nnew= " + newStr);

return newStr;
}

/*****
 * Get a attribute value in a tag
 *****/
function rmi_getTagAttribute(tag, key)
{
    var loc1 = tag.toLowerCase().indexOf(key);
    var loc2 = tag.indexOf("=", loc1) + 1;      // plus 1 for "="
    var first = loc2;
    var last = tag.length;

    if (loc1 == -1) return "";

    var whitespace_trimmed = false;
    for (var i=loc2; i<tag.length; ++i)
    {
        var aChar = tag.charAt(i);

        if (aChar != ' ' && ! whitespace_trimmed)
        {
            first = i;
            whitespace_trimmed = true;
        }

        if (aChar == ' ')
        {

```

```

        last=i;
        if (whitespace_trimmed) break
    }
}

if (first == -1)
    retTag = "";
else
    retTag = tag.substring(first, last);

if (rmi_JsDebug.indexOf(",rmi_getTagAttribute,") != -1)
{
    var msg = "key: " + key + "\n";
    msg += "old: " + tag + "\n";
    msg += "ret: " + retTag + "\n";
    msg += "first: " + first + "\n";
    msg += "last: " + last + "\n";
    alert("rmi_getTagAttribute:\n" + msg);
}

return retTag;
}

/*****
 * Set a new attribute value in a tag
 *****/
function rmi_setTagAttribute(tag, key, newval)
{
    var loc1 = tag.toLowerCase().indexOf(key);
    var loc2 = tag.indexOf("=", loc1) + 1;      // plus 1 for "="
    var first = loc2;
    var last = tag.length;

    if (loc1 == -1) return tag;

    var whitespace_trimmed = false;
    for (var i=loc2; i<tag.length; ++i)
    {
        var aChar = tag.charAt(i);

        if (aChar != ' ' && ! whitespace_trimmed)
        {
            first = i;
            whitespace_trimmed = true;
        }

        if (aChar == ' ')
        {
            last=i;
            if (whitespace_trimmed) break
        }
    }

    if (first == -1)
        retTag = tag;
    else

```

```

        retTag = tag.substring(0, first) + newval + tag.substring(last,
tag.length);

    if (rmi_JsDebug.indexOf(",rmi_setTagAttribute,") != -1)
    {
        var msg = "key: " + key + "\n";
        msg += "newval: " + newval + "\n";
        msg += "old: " + tag + "\n";
        msg += "new: " + retTag + "\n";
        alert("rmi_setTagAttribute:\n" + msg);
    }

    return retTag;
}

/*****
 * Handle the target within a tag in a frame wrapper mode
 *****/
function rmi_doTargetInFrameWrapperMode(tagStr)
{
    if (! rmi_FrameWrapperMode ) return tagStr;

    var retTag = tagStr;

    // ignore frames (will not be on top)

    if (rmi_startsWith(tagStr.toLowerCase(), "frame")) return retTag;

    if (rmi_UrlTarget == "_top")          // onTop & wrapper mode - force to encode
ALL
    {
        retTag = rmi_encodeTarget(tagStr, "href", true);
        retTag = rmi_encodeTarget(retTag, "action", true);
    }
    else          // encode only if target==_top
    {
        retTag = rmi_encodeTarget(tagStr, "href", false);
        retTag = rmi_encodeTarget(retTag, "action", false);
    }

    if (rmi_JsDebug.indexOf(",rmi_doTargetInFrameWrapperMode,") != -1)
        alert("rmi_doTargetInFrameWrapperMode:\n" + "old: " + tagStr + "\n" +
"new: " + retTag);

    return retTag;
}

/*****
 * Encode a target into a URL within a tag
 *****/
function rmi_encodeTarget(tagStr, key, force)
{
    var retTag = tagStr;

    var oldUrl = rmi_getTagAttribute(tagStr, key);

```

00650273 082900
0065280" 2205960

```
if ( rmi_endsExactlyWith(oldUrl, rmi_Vars) )           // already encoded
    return retTag

if (! rmi_startsWith(oldUrl, rmi_ProxyURL) &&
    ! rmi_startsWith(oldUrl, rmi_SecureProxyURL) )
    return retTag                                     // not translated (e.g.
gif)

if (force)      // force to rewrite
{
    retTag = rmi_setTagAttribute(tagStr, key, oldUrl + rmi_Vars);
}
else
{
    // If target==_top
    var targetVal = rmi_getTagAttribute(tagStr, "target");
    targetVal = rmi_trimQuotes(targetVal);

    if (targetVal == "_top")
    {
        retTag = rmi_setTagAttribute(tagStr, key, oldUrl + rmi_Vars);
    }
}

if (rmi_JsDebug.indexOf(",rmi_encodeTarget,") != -1)
    alert("rmi_encodeTarget\n" + "old: " + tagStr + "\n" + "new: " +
retTag);

return retTag;
}

/*****
 * Append a string to a URL if no such string at the end yet
 *****/
function rmi_appendToUrl(url, str)
{
    var urlStr = "" + url;
    var ret;

    if ( rmi_endsExactlyWith(urlStr, str) )           // See
/rmivars%3f...
        ret = urlStr;
    else if ( rmi_endsExactlyWith(urlStr, unescape(str) ) ) // See
/rmivars?...
    {
        var array = urlStr.split(unescape(str));
        ret = array[0] + str;
    }
    else
        ret = urlStr + str;

    return ret;
}

/*****
 * Collapse a path (i.e. remove parts of a path like "dir/..")
 *****/
```

```

*****/
function pathCollapse(path)
{
    var slist = path.split("/");
    var stack = new Array();
    var counter = 0;

    for (var i = 1; i < slist.length; ++i)
    {
        var item = slist[i];

        if (item != "..")
            stack[counter++] = item;
        else if (counter > 0)
            --counter;
    }

    stack.length = counter;

    //alert("mpath " + path + "\nmpath " + "/" + stack.join("/"));
    return ("/" + stack.join("/"));
}

/*****
 * Translate a string, then do eval().
 *****/
function rmi_eval(code)
{
    return eval(code);
}

/*****
 * This function will be overridden at run time if necessary
 *****/
function rmi_xjs(code)
{
    return code;
}

/*****
 * Translate a string, then do setTimeout().
 *****/
function rmi_setTimeout(code, msec)
{
    return setTimeout(code, msec);
}

/*****
 * Get RMI cookies
 *****/
function rmi_getCookie(cookie)
{
    // alert("rmi_getCookie:\n" + "cookies: " + cookie + "\nrmi_cookies: " +
    rmi_CurrentCookies);

    if (typeof rmi_CurrentCookies == "undefined")
        return "";
}

```

00650273.082900
0065280" 2205960

```
    else
        return rmi_CurrentCookies;
}

/*****
 * Set RMI cookies
 *****/
function rmi_setCookie(cookieLHS, cookieRHS)
{
    // Set RMI cookie @ the server side

    var serverCookie = rmi_xlateServerCookie( cookieRHS );
    if (serverCookie == "") return;

    var newCookieTail = "path=/rmi; domain=" + rmi_CookieDomain;
    var newCookie = "rmiCookie" + (new Date()).getTime() + "=" +
        escape(serverCookie) + "; " + newCookieTail;

    document.cookie = newCookie;

    // Set rmi_CurrentCookies @ the client side

    var clientCookie = rmi_xlateClientCookie( cookieRHS );
    if (clientCookie == "") return;

    if (typeof rmi_CurrentCookies == "undefined")
        rmi_CurrentCookies = clientCookie;
    else
        rmi_CurrentCookies += ";" + clientCookie;
}

/*****
 * Verify cookie's domain (before setting the cookie)
 *****/
function rmi_verifyCookieDomain(domain)
{
    var hostname = rmi_getOriginal(window.location, 'hostname');

    if (rmi_endsExactlyWith(hostname.toLowerCase(), domain.toLowerCase()) )
        return true;
    else
        return false;
}

/*****
 * Parse a cookie string, returns a new client cookie
 * (for browsers) without path, domain, expires, & secure fields.
 *****/
function rmi_xlateClientCookie(cookieStr)
{
    var list = cookieStr.split(";");
    var ret = "";

    for (var i = 0; i < list.length; ++i)
    {
        // NOTE: array's length > 2 if there are more than 2 '='
    }
}
```

```

var array = list[i].split("=");

if (array.length < 1) continue;

var key = rmi_trim(array[0]).toLowerCase();

// Verify the cookie domain if there

if (key == "domain")
{
    var domainVal = rmi_trim(array[1]).toLowerCase();
    if ( rmi_verifyCookieDomain(domainVal) )
    {
        continue;           // OK
    }
    else
    {
        ret = "";
        break;
    }
}

if (key == "path") continue;
if (key == "expires") continue;
if (key == "secure") continue;

if (i != 0) ret += ";";
ret += array.join("=")
}

return ret;
}

/*****
 * Parse a cookie string, returns a new server cookie
 * (for rmi proxy server) with path & domain (if not there originally).
 *****/
function rmi_xlateServerCookie(cookieStr)
{
    var list = cookieStr.split(";");
    var ret = "";
    var hasDomain = false;
    var hasPath = false;

    for (var i = 0; i < list.length; ++i)
    {
        // NOTE: array's length > 2 if there are more than 2 '='

        var array = list[i].split("=");

        if (array.length < 1) continue;

        var key = rmi_trim(array[0]).toLowerCase();

        if (key == "domain")
        {
            hasDomain = true;

```

```

        var domainVal = rmi_trim(array[1]).toLowerCase();
        if ( ! rmi_verifyCookieDomain(domainVal) )
            return ""; // BAD domain - RETURN
    }

    if (key == "path") hasPath = true;

    if (i != 0) ret += ";";
    ret += array.join("=")
}

if (! hasDomain) ret += "; domain=" + rmi_Hostname;
if (! hasPath) ret += "; path=" + rmi_Pathname;

return ret;
}

//.-->

```

006280" E2205960